A System Level Energy Model for HPL-PD Microarchitecture in Trimaran framework (TRIREME)

Rajeshwari Banakar, Mongkol Ekpanyopang, Kiran Puttaswamy, Rodric Rabbah,
M. Balakrishnan, Vincent Mooney III, Krishna Palem
banakar, mbala@cse.iitd.ernet.in, Indian Institute of Technology, Delhi 110 016
kiranp, rabbah,mooney, palem@ece.gatech.edu, Georgia Institute of Technology Atlanta, GA 30332

ABSTRACT

In this paper a system level energy model called TRIREME, is presented for HPL-PD microarchitecture which is used in Trimaran Compiler framework studies. The number of accesses for the various computational units are obtained from the trimaran framework, which gives the performance estimates also. We focus on the details of the HPL PD at the microarchitectural level and how the energy models for the processor are constructed. Our system level energy model can be used in Trimaran framework for energy-aware computing, to validate the compiler techniques for the benefits of energy saving due to introduction of a specific compiler optimization.

1. INTRODUCTION

Power dissipation in hand embedded systems is an is important concern for the architects at the system level. The detailed architecture of such systems is very much application driven to meet power and performance requirements. It is reported that 80% of the power analysis can be performed at the architectural level [3]. We build a system level energy estimation methodology for HPL PD architecture. This will assist the Trimaran framework research community to validate their application specific studies and various compiler techniques for energy efficiency.

HPL PD is a parametric processor architecture used for research. Its main purpose is to study the benefits of the compiler technology, where there is a need to exploit such architectures [2]. Up to now HPL PD was defined at the architectural level for research purpose. Since our focus is to develop a energy estimation methodology for this architecture, we define HPL PD at the microarchitecture level.

We emphasize that the main purpose of this HPL PD microarchitecture model - TRIREME defined by us is to serve as a research tool. Our strategy is the first approach which combines the synthesized models and analytical models to get estimate of the system level energy consumption.

In order to develop the HPL PD microarchitectural details and the processor energy model we use the Verilog models for the various components. These components are synthesized using Synopsys Design compiler. Synopsys Power Compiler is used to build the energy database for the processor [21]. Although the verilog approach of power modeling is a time consuming task, it is a one time job to build the database and use it throughout for the experimental analysis and is reasonably accurate. The cache energy database for a given configuration is built using the CACTI tool [4].

Several interesting results are obtained from our approach.

- We could identify the hot spots of the energy consuming units. This gives the system architect an opportunity to fine tune the processor core at the component level.
- The on-chip memory energy estimates for the L1 and the L2 cache, assist the designer at the algorithmic level to study the impact on energy, by varying the cache configuration parameters, like the cache size, associativity and the block size.
- We include the system level processor core energy estimates for the stall cycles in the processor. The stall cycles depend on the processor microarchitecture, and the energy estimates of this, will be essential in validating the compiler techniques like speculation etc.

This model is used to estimate the energy estimations for various application benchmarks. The performance estimates are obtained from the Trimaran framework through simulation. The performance and energy results for various benchmarks use TRIREME. The total system energy spent and the fraction of energy consumed in the CPU, L1 cache and L2 cache are identified.

Related Work

Power analysis can be done at various abstraction levels [14] from architectural level to spice level. We studied various approaches followed by other power estimation models. In Wattch [6], a power analyzer for superscalar architecture, the functional unit description of alu, multiplier etc are a black box, to the user.

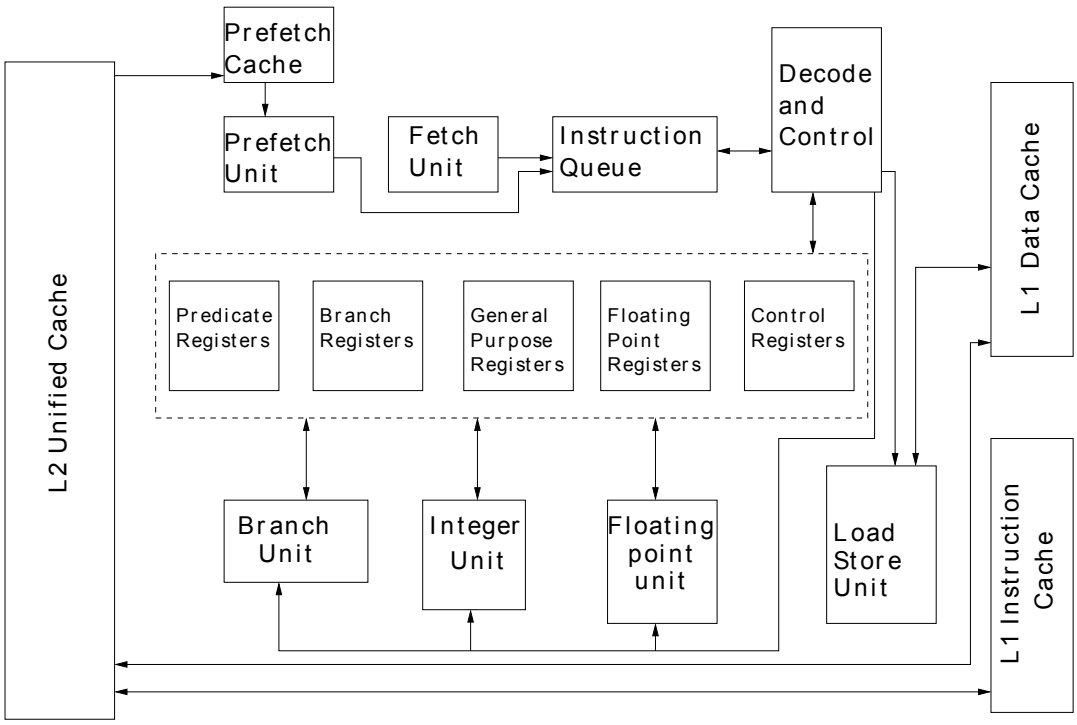

Figure 1: HPL-PD microarchitecture - A schematic diagram

Rita et al. [20] address the issue of architectural level power estimation using switching activity approach for the processor. In [15] the power estimates are based on the instruction level modeling for PR1900 processor architecture. Kavel et al. address the high level power estimation with interconnect effects. Dake Liu and et al. [17] develop the analytical models for the various parts of the processor and validate their technique for ALPHA 21604 processor and the INTEL 80386 processor core. They conduct various experiments at the transistor level to obtain the gate capacitance statistics, area of the transistor and other technology dependent parameters. Recently, interest has been focused on studying the benefits of compiler techniques for energy savings and improving the performance. In a report, [13] the energy benefits of data-remapping are presented for the ARM model.

Background

System level power corresponds to the following sub units : the processor core, bus power, on-chip memory power, interconnect power, clock distribution, and input output pads. Various abstraction levels used to compute power are transistor (spice simulation), gate level simulation, RTL level and the behavioral level estimations. We can identify two main approaches for system level power estimates mainly

- Hardware approach which may be analytical based [17], microarchitecture based or data sheet approach. Microarchitecture approach may again be model based [20], synthesis based and simulation approach.
- Software approach is based on actual measurements and obtaining the power values for various instructions based on Vivek Tewari's model [10]. Alternative to this is to build the instruction level model and then obtain the instruction level power estimates for the processor under study [15]

The rest of the paper is organized as follows: Section 2 describes the HPL PD microarchitecture details and the energy models. In section 3 we give an overview of the work flow that we have used to develop the system level energy. In section 4 we explain the computation of base energy for the processor and cache. In section 5 we present the experimental results for various benchmarks, while we conclude in section 6 with some ideas on future work.

2. HPL-PD MICROARCHITECTURE

Before explaining the HPL PD microarchitecture, we describe how an instruction is executed in the processor. This essentially depicts the necessity of various modules in the microarchitecture. The various phases to execute an instruction that loads data from memory into a register has the following stages,

- Fetching instruction from memory.
- Decoding the instruction, fetching any operands it uses from the register file (part of the address etc)
- Loading data from the memory.
- Writing data to the register file.

Fig 1 shows the microarchitecture of the HPL-PD microarchitecture. The processor has three main components

- Control path.
- Data path.
- On-chip memory.

The definition of the microarchitecture provides the various models to be used at the component level for energy estimation. In detailed terms, the HPL-PD microarchitecture defines how the processor interfaces with the memory resources. The top level of the microarchitecture consists of different digital functional blocks and memory modules used for parallel execution. The control path unit mainly has the instruction fetch unit, the prefetch unit, instruction queue, the decode and control logic unit, and the load store unit. The data path unit consists of various type of registers, functional unit with integer unit, floating point unit and branch unit.The different types of registers in HPL PD are general purpose registers, floating point registers, predicate registers and branch target registers. The HPL PD has two levels of cache viz L1 cache and the L2 cache used for large memory management with 64 bit addressing. L2 cache is unified, while L1 has separate Instruction and data cache.

HPL PD is a high performance processor used in the trimaran compiler framework.

3. WORK FLOW IN SYSTEM LEVEL ENERGY ESTIMATION

In this section we explain the work flow used in our approach to build the system level energy model, using the HPL-PD microarchitecture defined in the previous section. The method presented here assumes a fixed architecture for the processor, and a energy database is built for the entire processor, which consists of the energy consumed for each of the units. We start by illustrating the estimation process using a C application benchmark. The work flow of the energy estimation using the energy database of the processor modules is shown in Fig. 2. The block shown in dotted line (to build the energy database) is done only once for the microarchitecture defined. The work flow can be distinctly identified as two separate tasks. One is obtaining the performance estimates from the Trimaran framework for a given application. Second, is the energy estimation of the processor and the on-chip memory, which uses the number of accesses for each functional unit from the Trimaran simulator.

The HPL-PD architecture is defined at the component level to compute the overall energy. To accomplish this, we use the verilog designs for each of the modules. The overview of how the energy estimates are computed is given in Fig 3

The synthesis is done using the Synopsys Design Compiler using the TSMC 0.25 um technology.

Integer Unit

For example, we describe a typical integer ALU model which can perform integer arithmetic operations (addition and subtraction) and logical operations (and, or, not) in Verilog HDL. Then, we synthesize the RTL description of the Integer ALU unit using Synopsys Design Compiler [21] targeted towards the TSMC 0.25μ standard cell library from LEDA Systems, Inc. [22].

Power Compiler

A power analysis tool called Power Compiler from Synopsys, Inc. is used to obtain the energy estimates. Power Compiler computes average power consumption in terms of switching power, internal power and static power based on the switching activity of the nets in the design and also the technology parameters of the standard cell library. It also gives a hierarchical percentage breakdown of the power consumed by individual modules in the design.

The inputs to Power Compiler are as follows:

- Gate-level net list
- Technology library
- Design constraints
- Capacitance parasitics
- Switching activity of nets.

The sequence of steps for power analysis flow is shown below:

- Input the gate-level netlist to Power Compiler.
- Set up the Technology library.
- Specify the constraints on design like clock period, rise and fall time and clock skew.
- Specify the statistical wire load models for capacitance/resistance parasitics of the interconnecting nets.
- Annotate the switching activity of nets onto the design.
- Report the power estimates for dynamic and static sower.

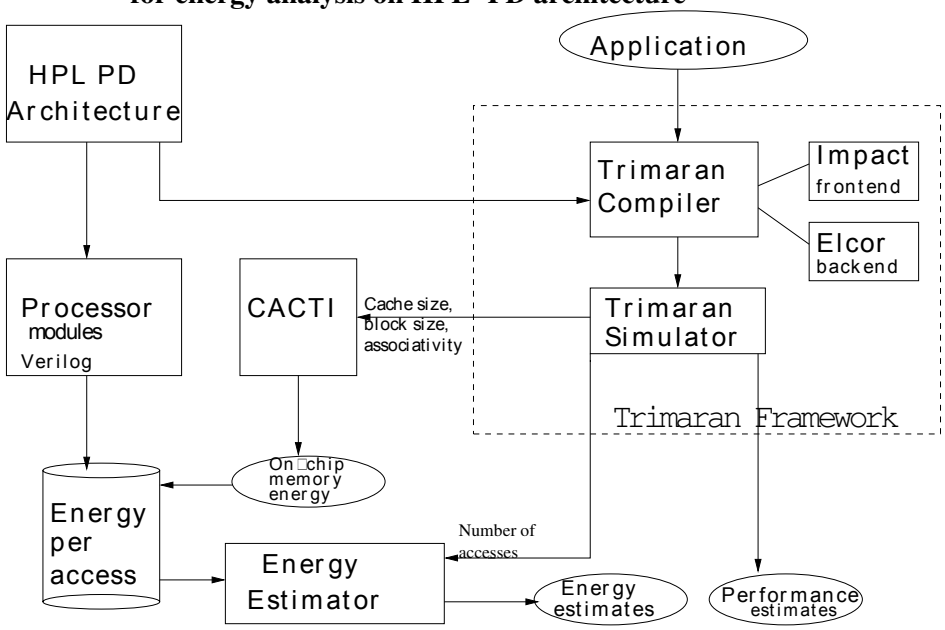

Figure 2: An overview of the System level energy estimation work flow

The system level energy estimation is hardware based, using which we can identify the most energy consuming parts of the system. In order to integrate this to the Trimaran framework, we build the processor energy database. We multiply these energy estimates from the energy database with the access counts of the respective units to get the energy estimates. This part is automated and integrated into the work flow using the trimaran framework.

Trimaran Framework

Trimaran is an integrated compilation and performance-monitoring infrastructure. Trimaran compiler framework is built using Explicitly Parallel Instruction Computing architecture called HPL PD. HPL-PD is a parameterized architecture which supports predication, compiler controlled management of the memory hierarchy, and control and data speculation. HPL-PD, trimaran compiler can also generate the code for other architecture such as StrongArm. HPL-PD machine is parameterized architecture used in trimaran framework. It is possible to modify the number of functional units, register files and memory size by changing machine description language called HMDES. The system also supports cycle-accurate simulation and flexible performance monitoring environment. By integrating Dinero which is a cache simulator, with Trimaran system, cache performance can be monitored.

The Trimaran compiler infrastructure consists of the following components.

- A machine description facility HMDES to characterize the target processor
- A compiler front end, called IMPACT for high-level optimization. IMPACT performs parsing, type checking and a large number of machine independent optimization's
- A compiler back end, called Elcor, parameterized by HMDES. Elcor performs machine dependent analysis and optimization, instruction scheduling and register allocation.
- An intermediate language called Rebel, which has an internal and textual representation with conversion routines between the two. The intermediate language is a sophisticated IR capable of capturing control flow, data and control dependence among other things.
- A cycle accurate and extendible simulator parameterized by MDES for performance studies.
- A GUI to configure and run Trimaran system.

4. BASE ENERGY COMPUTATION

We define the system level energy for the HPL PD microarchitecture as the base energy. In this section we present how the base energy computation is done.

System energy

From the HPL-PD microarchitecture defined in section 2 we identify the various units in the processor core. The number of times each unit is accessed is important, since the more often the unit is activated, the more energy it will consume. The total energy estimate of the processor core and the memory hierarchy is defined as the system energy E_{sys}.

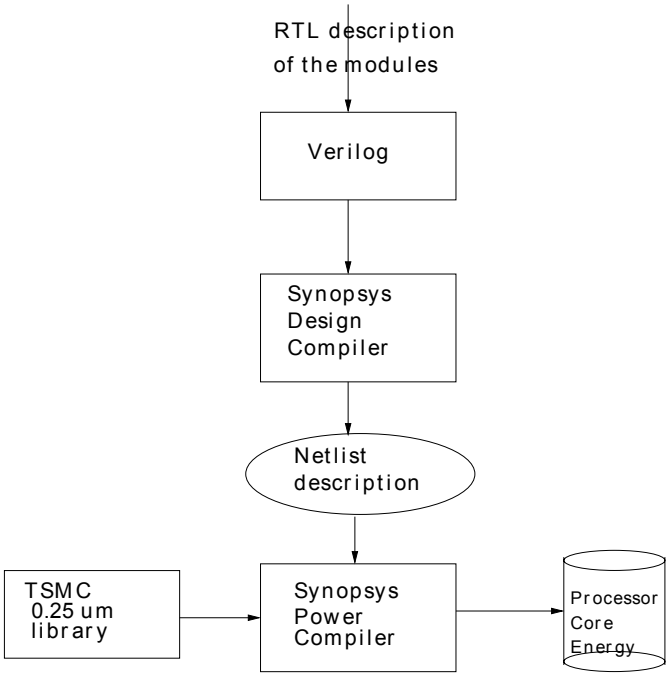

Figure 3: Overview of the processor energy estimation

$$E_{sys} = E_{cpu} + E_{L1} + E_{L2} + E_{stall} \quad (1)$$

where E_{cpu} is the energy spent in the processor core
E_{L1} is the energy spent in the L1 cache.
E_{L2} is the energy spent in the L2 cache.
E_{stall} is the energy consumed due to stall cycles.

Processor core energy
To evaluate the system level energy we need the processor core energy consumption. The RTL level description of the processor units like the alus, floating point units, load store units, registers are developed, to obtain the processor energy estimate database (refer Fig 1).

$$E_{cpu} = \sum N_{unit} * E_{unit} \quad (2)$$

where E_{cpu} is the processor energy
N_{unit} is the number of times the unit is accessed.
E_{unit} is the energy per access of the unit.

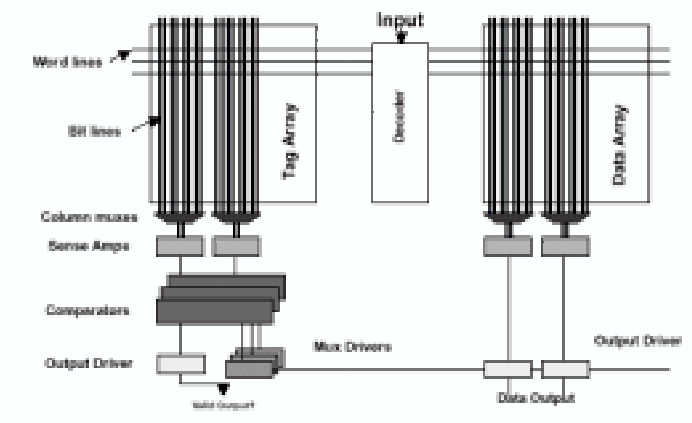

Figure 4: Cache Memory organization [4]

Stall Cycles
If a resource is required and none is available, the instruction execution does not proceed until the required resource is available, which is the cause of stall cycles. Such stalls in the simulation environment, will not activate all the processor core units. Only a few components of the processor are activated. Based on this reasoning, we adopt the inactive power to be one percent of the active power of the system. It is evident that the cpu energy due to stall cycles will be less than the processor energy during the dynamic cycles. We assume that each module has an active power consumption that depends on the switching activity (active power consumption). Inactive power consumption consists of static/leakage power consumption where the switching activity is negligibly small.
The energy due to stall cycles is defined as E_{stall} and used in equation 1 to compute the system level energy estimates.

Cache Energy estimates
Analytical model is used to estimate the cache energy using cacti [4] tool. Analytical methods attempt to relate the energy consumption of a particular description to fundamental quantities that describe the physical capacitance, the transistor parameters, the technology (feature size) and the switching activity of the design. An overall fixed switching activity of 0.5 is assumed in this model. The advantage of using the cacti tool is that, it accepts the cache size, block size, associativity and feature size as input, and gives the energy estimates along with the break ups of the tag array, data array, sense amplifier and other peripheral components as shown in Fig. 4 which gives the cache organization [4].
The configuration uses six transistor SRAM cell as the basic memory cell in the tag array and the data array units. A detailed explanation can be found in [4, 5, 11]. The energy is consumed in the tag array unit, data array unit, decoder unit, peripheral, word line and bit line units. The C application program is fed to the Trimaran framework and the cache parameters are identified. The cache configuration parameters like the cache size, associativity, block size are obtained from Trimaran simulator for the application under consideration.

5. EXPERIMENTAL RESULTS

To investigate the energy model developed for the HPL PD microarchitecture several experiments were conducted using a few SPEC2000 benchmarks [18], and other illustrative benchmarks, which are a collection of compute-intensive programs, (integer and floating type) used to evaluate the performance of a computer's CPU, memory system, and compilers. The results obtained were then used to evaluate the behavior of the benchmarks for mainly two specific

Benchmark	Energy estimates % CPU	L1 cache	L2 cache	Performance estimates % Compute cycles	Stall Cycles
130.li	40	20	40	92	8
132.ijpeg	35	26	38	98.7	1.3
134.perl	34	22	44	87	13
164.gzip	39	24	37	99.5	0.5
179.art	40	15	45	77	23
181.mcf	45	14	41	71	29
183.equake	46	17	37	65	35
188.ammp	29	25	46	74	26
197.parser	47	20	33	87.3	12.7
300.twolf	68	14	18	99.7	0.3
DM	50	19	31	99.9	0.1
RT	42	25	33	99.8	0.2
Transitive	46	22	32	54.5	45.5
bh	30	24	46	81	19
em3d	35	3	62	90	10
Neighborhood	85	5	10	99	1
Pointer	80	16	4	99.5	0.5

Table 1: Experimental results

issues. One is the performance estimates, which includes the stall cycles. Second is the amount of energy spent in the cpu, L1 cache and the L2 cache.

Performance estimates

For clarity in presenting key observations, consider first the performance estimates of benchmarks presented in table 1. This table gives the total dynamic cycles and the stall cycles as a function of the size of the register files, the size of L1 data cache, the size of L1 instruction cache, the associativity of the L1 data and instruction cache, the number of functional units, the size of L2 cache with block size and its associativity. One observation is that if stall cycles are modeled in the system level energy estimation, there will be a overall change in cpu energy, with this change being larger for benchmarks having more stall cycles. If the stall cycles energy estimation is not done, unnecessarily one gets higher energy estimates. Although other analysis can be done using the trimaran framework, we restrict ourselves to the baseline energy estimates of the HPL-PD microarchitecture which is our main goal in this work. In other words we are interested in studying the benchmarks for energy distribution mainly in the CPU, L1 cache and the L2 cache.

The values presented in the table are sensitive to the number of physical registers, the size of L1 cache and L2 cache. If the number is increased, there is a change in the total dynamic compute cycles and the stall cycles. The stall cycles vary from 0.5% to 45.5%.

Energy estimates

The energy estimates for various benchmarks are depicted in table 1. The results presented illustrate the analysis of the benchmark for energy distribution among cpu, L1 cache and L2 cache. For example Neighbourhood benchmark shows 85% of energy consumption in the cpu, which clearly indicates that it is not memory intensive application. An important observation suggested by the data presented in this table, indicate the ability to identify the hotspots, that is which are the energy consuming units in the baseline HPL PD microarchitecture for the given application. Since the breakup of the cpu energy is also available to us, we can identify the energy consumed by each sub unit and evaluate the most energy hungry unit. Consider the 183.equake benchmark which is used to obtain the seismic analysis statistics for earthquake simulations. The cpu energy spent is 46%, with 17% energy in L1 cache and 37% energy in the L2 cache. The spec2000 benchmark suite has an image processing application, namely the scanner module in the 179.art software. This application illustrates training of sequences for neural networks in scanning the images. Here we observe that cpu energy is 40% with L1 cache energy of 15%. The result for the 188.ammp benchmark, which is a molecular mechanics program depicts the 29% energy consumption in cpu. This memory intensive application computes the torsion, velocity, angle bond correlation record and a number of other parameters. The results illustrate the basic behavior of the application, indicating 71% of the energy consumption is in the on-chip memory hierarchy. From the table we observe that the variation for the benchmarks selected is 29% to 85% in CPU energy, 3% to 26% in L1 cache, 4% to 62% in L2 cache.

6. CONCLUSIONS AND FUTURE WORK

We have presented the microarchitecture for the HPL PD processor which is used in trimaran framework. A systematic workflow is developed to evaluate the distribution of energy in processor core, L1 cache and L2 cache. We model the energy consumption due to stall cycles also. We develop a system level energy management procedure for the trimaran framework which is popularly used to study the effects of compiler techniques. Our approach helps in early design space exploration and aids the designer to

- Use suitable compiler techniques to reduce the on-chip energy requirement.
- Identifying critical regions of the architecture which

contribute to major energy consumption, experiment some parameterized configurations to reduce the energy consumption choosing configuration defined as the optimum, which consumes less energy.

An interesting future direction that we see, from this work is to study the interconnect energy estimates and the clock distribution energy estimates for HPL PD microarchitecture. This will increase the accuracy in the system energy estimates.

7. REFERENCES

[1] Peter Markstein, *IA-64 and Elementary Functions, Speed and Functions*, Prentice Hall PTR, New Jersey, 2000.

[2] Vinod Kathail, Schlansker M S, B Ramakrishna Rau : *HPL-PD Architecture Specification: Version 1.1*, Technical Report HPL-93-80, February 2000.

[3] Sheng Deng : *Estimating IC power consumption at the RT level*, In EDA IC Analysis Electronics Engineer, October 1999.

[4] S Wilton and Norm Jouppi : *Cacti : An enhanced access and cycle time model*, IEEE Journal of Solid State Circuits, May 1996.

[5] Rajeshwari Banakar, S Steinke, B S Lee, M Balakrishnan and P Marwedel, *Comparison of cache and scratch pad based memory system with respect to performance, area and energy consumption*, Technical Report 762, University of Dortmund, Sep 2001.

[6] D Brooks, V. Tewari, and M Martonosi : *Wattch : A framework for architectural-level power analysis and optimizations*, In 27th Annual International Symposium on Computer Architecture, pages 83-94, Association for Computing Machinery, June 2000.

[7] M Kamble and K. Ghose : *Modeling Energy Dissipation in Low Power Caches*, Technical Report, CS-TR-98-02, Department of Computer Science, SUNY-Binghamton, 1998.

[8] J Rabaey : *Digital Integrated Circuits*, PHI Publications, Prentice Hall, Inc, 1996.

[9] Triceps, http://www.trimaran.org/triceps.html.

[10] V Tiwari, S Malik, A Wolfe and M C Lee : *Instruction level power analysis and optimization of software*, In Journal of VLSI Signal Processing, Kluwer Academic Publishers, August/Sept 1996.

[11] Rajeshwari Banakar, S Steinke, B S Lee, M Balakrishnan and P Marwedel, *Scratchpad memory: A design alternative to on-chip caches in embedded systems*, In the Proceedings of Hardware Software Co design Symposium CODES 2002, Estes Park Colorado, May 2002.

[12] Krishna V Palem, Rodric M Rabbah, Vincent Mooney III, Pinar Korkmaz and Kiran Puttaswamy : *Design Space Optimization of Embedded Memory Systems via Data Remapping*, Technical Report, GIT-CC-020011, Georgia Institute of Technology, February 2002.

[13] Pinar Korkmaz, Kiran Puttaswamy, Vincent Mooney III : *Energy Modeling of a Processor core using Synopsys and of memory hierarchy using Kamble and Ghose model*, Technical Report, CREST-TR-02-002, Center for Research on Embedded Systems and Technology, Georgia Tech, Atlanta USA, February 2002.

[14] Rajeshwari M Banakar, Ranjan Bose, M Balakrishnan : *Low power design - Abstraction levels and RTL design techniques*, VLSI test and design Workshop, VDAT 2001 Bangalore, Aug 2001

[15] Akshaye Sama, M Balakrishnan, J F M Theeuwen : *Speeding up Power Estimation of Embedded Software* In the Proceedings of ISPLED 2000, Rapallo-Portacino Coast, Italy, July 2000.

[16] http://www.hardwaresecrets.com/ia64.html.

[17] Dake Liu, Christer Svensson : *Power Consumption Estimation in CMOS VLSI Chips*, IEEE Journal of Solid State Circuits, Vol 29, No 6, June 1994.

[18] http://www.spec.org/osg/cpu2000.

[19] Neil H E Weste, Kamran Eshraghian : *Principles of CMOS VLSI design - A systems perspective*, Second Edition, Addison Wesley Publication, 1998.

[20] Rita Yu Chen, Mary Jane Irwin : *Architecture-Level Power Estimation and Design Experiments*, ACM Transactions on Design Automation of Electronic Systems, Vol 6, No 1, January 2001, Pages 50-66.

[21] Synopsys, Inc., *http://www.synopsys.com*

[22] LEDA Systems, Inc., *http://www.ledasys.com*